Práctica 4 SWAP

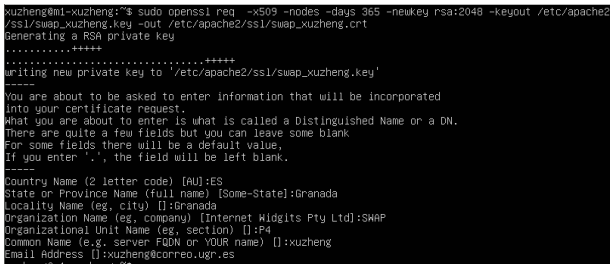
XuSheng Zheng

Índice

1. Certificado autoafirmado SSL

Empezamos habilitando el módulo SSL de Apache y creando el directorio para ubicar los certificados:

Ahora creamos el certificado con los siguientes datos:



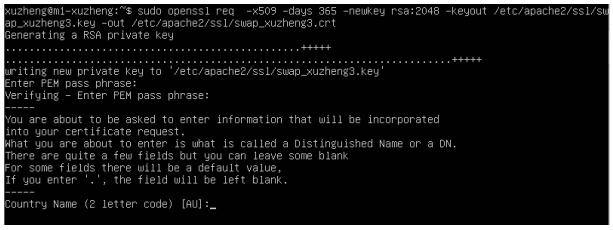
1.1. Opciones avanzadas

De las opciones introducidas anteriormente, podemos prescindir de **nodes** para añadir una contraseña a la clave. Además, tenemos las siguientes opciones que pueden ser interesantes:

- **-config filename**: permite especificar un archivo de configuración para la creación de los certificados.
- **-subj arg**: permite especificar datos para la creación del certificado. El argumento debe ser de la forma **/type0=value0/type1=value1/type2=....**. Al introducir esta opción no nos pedirá los datos como habíamos hecho.
- **-addext ext**: permite añadir una extensión de x509 en el certificado generado.

Como ejemplo creamos el siguiente certificado:

Podemos ver que nos pide la contraseña para generar el certificado. Si eliminamos la opción **-subj** podemos ver que aparte de pedirnos la contraseña nos pide también el resto de informaciones:



2. Apache con certificado SSL

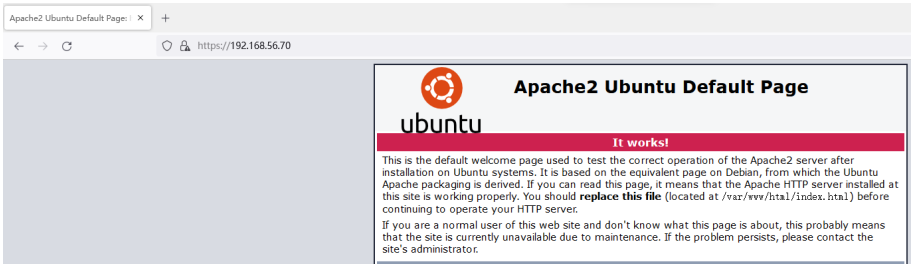
Editamos el archivo de configuración */etc/apache2/sites-available/default-ssl.conf*:

```
#   Enable/Disable SSL for this virtual host.
SSLEngine on

SSLCertificateFile /etc/apache2/ssl/swap_xuzheng.crt
SSLCertificateKeyFile /etc/apache2/ssl/swap_xuzheng.key

#   A self-signed (snakeoil) certificate can be created by installing
#   the ssl-cert package. See
#   /usr/share/doc/apache2/README.Debian.gz for more info.
#   If both key and certificate are stored in the same file, only the
#   SSLCertificateFile directive is needed.
#SSLCertificateFile    /etc/ssl/certs/ssl-cert-snakeoil.pem
#SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

#   Server Certificate Chain:
#   Point SSLCertificateChainFile at a file containing the
#   concatenation of PEM encoded CA certificates which form the
#   certificate chain for the server certificate. Alternatively
#   the referenced file can be the same as SSLCertificateFile
#   when the CA certificates are directly appended to the server
#   certificate for convinience.
#SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt
```

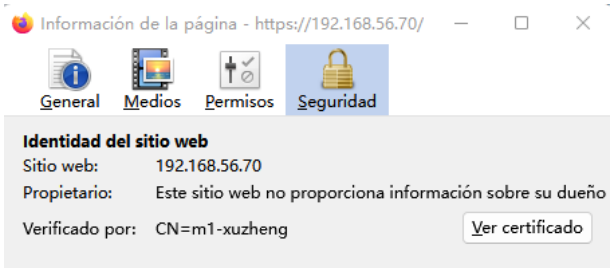
Activamos **default-ssl** y reiniciamos Apache:

```
xuzheng@m1-xuzheng:~$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
xuzheng@m1-xuzheng:~$ sudo service apache2 reload
```

Para comprobar que se ha instalado correctamente el certificado accedemos desde el navegador del anfitrión:



Accedemos al certificado desde el candado con exclamación a la izquierda de la URL:



Ahora procedemos a copiar los archivos que hemos creado anteriormente a m2 y m3:

Al igual que en m1, en m2 creamos el directorio */etc/apache2/ssl*, copiamos los archivos en dicho directorio y configuramos **default-ssl**:

```
xuzheng@m2-xuzheng:~$ sudo mkdir /etc/apache2/ssl
[sudo] password for xuzheng:
xuzheng@m2-xuzheng:~$ sudo mv swap_xuzheng.* /etc/apache2/ssl/
xuzheng@m2-xuzheng:~$ sudo a2enmod ssl & sudo service apache2 restart
[1] 1712
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certifica
tes.
To activate the new configuration, you need to run:
  systemctl restart apache2
[1]+  Done                    sudo a2enmod ssl
xuzheng@m2-xuzheng:~$
```

```
#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

SSLCertificateFile /etc/apache2/ssl/swap_xuzheng.crt
SSLCertificateKeyFile /etc/apache2/ssl/swap_xuzheng.key
```

```
xuzheng@m2-xuzheng:~$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
xuzheng@m2-xuzheng:~$ sudo service apache2 reload
xuzheng@m2-xuzheng:~$ _
```

Podemos comprobar con **cURL** que m2 acepta peticiones HTTPS:

```
txd16@DESKTOP-S58Q8SA MINGW64 /e/DGIIM/QUINTO 2º CUAT/SWAP/Prácticas/P4 (main)
$ curl -k https://192.168.56.71/swap.html
<HTML>
<BODY>
SWAP M2
Web de ejemplo de xuzheng para SWAP
Email:xuzheng@correo.ugr.es
</BODY>
</HTML>
```

3. Nginx como balanceador para peticiones HTTPS

En m3 activamos **Nginx** y aseguramos de que esté en funcionamiento:

```
xuzheng@m3-xuzheng:~$ sudo systemctl enable nginx.service
Synchronizing state of nginx.service with SysV service script with /lib/systemd/systemd-sysv-install
.
Executing: /lib/systemd/systemd-sysv-install enable nginx
xuzheng@m3-xuzheng:~$ sudo systemctl start nginx.service
xuzheng@m3-xuzheng:~$ sudo systemctl status nginx.service
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-04-22 17:04:25 UTC; 4s ago
       Docs: man:nginx(8)
    Process: 13400 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/S
    Process: 13388 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, s
   Main PID: 13404 (nginx)
      Tasks: 2 (limit: 4653)
     CGroup: /system.slice/nginx.service
             ├─13404 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
             └─13406 nginx: worker process

abr 22 17:04:25 m3-xuzheng systemd[1]: Starting A high performance web server and a reverse proxy se
abr 22 17:04:25 m3-xuzheng systemd[1]: nginx.service: Failed to parse PID from file /run/nginx.pid:
abr 22 17:04:25 m3-xuzheng systemd[1]: Started A high performance web server and a reverse proxy ser
lines 1-15/15 (END)_
```

Creamos un nuevo directorio *ssl* para guardar los archivos:

```
xuzheng@m3-xuzheng:~$ sudo mkdir ssl
xuzheng@m3-xuzheng:~$ sudo mv swap_xuzheng.* ssl/
xuzheng@m3-xuzheng:~$ _
```

Editamos el archivo */etc/nginx/conf.d/default.conf* para añadir un nuevo bloque **server** con los siguientes datos:

```
        location /
        {
                proxy_pass http://balanceo_xuzheng;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_http_version 1.1;
                proxy_set_header Connection "";

        }
}

server{
        listen 443 ssl;
        ssl on;
        ssl_certificate         /home/xuzheng/ssl/swap_xuzheng.crt;
        ssl_certificate_key     /home/xuzheng/ssl/swap_xuzheng.key;
        server_name balanceador_xuzheng;

        access_log /var/log/nginx/balanceador_xuzheng.access.log;
        error_log /var/log/nginx/balanceador_xuzheng.error.log;
        root /var/www;

        location /
        {
                proxy_pass http://balanceo_xuzheng;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_http_version 1.1;
                proxy_set_header Connection "";
        }
}
```

Reiniciamos **Nginx** y comprobamos desde el anfitrión:

```
txd16@DESKTOP-S58Q8SA MINGW64 /e/DGIIM/QUINTO 2º CUAT/SWAP/Prácticas/P4 (main)
$ curl -k https://192.168.56.72/swap.html
<HTML>
<BODY>
SWAP M1
Web de ejemplo de xuzheng para SWAP
Email:xuzheng@correo.ugr.es
</BODY>
</HTML>

txd16@DESKTOP-S58Q8SA MINGW64 /e/DGIIM/QUINTO 2º CUAT/SWAP/Prácticas/P4 (main)
$ curl -k https://192.168.56.72/swap.html
<HTML>
<BODY>
SWAP M2
Web de ejemplo de xuzheng para SWAP
Email:xuzheng@correo.ugr.es
</BODY>
</HTML>
```

3.1. Opciones avanzadas

A partir de la configuración anterior podemos añadir algunas mejoras. Puesto que las operaciones SSL consume recursos extras del CPU, existen dos maneras de minimizar el número de estas operaciones por cliente: la primera es habilitar la opción **keepalive** para poder mandar varias peticiones por conexión y la segunda es mediante la reutilización de parámetros de sesión SSL para evitar **ssl handshakes**. Dichas sesiones se almacenan en un cache que se configura con la directiva **ssl_session_cache**. El timeout por defecto de dicho caché es de 5 minutos, para modificarlo podemos utilizar la directiva **ssl_session_timeout**.

En el siguiente ejemplo habilitamos un cache de tamaño 1MB con un timeout de 10 minutos:

```
server{
        listen 443 ssl;
        ssl on;
        ssl_certificate         /home/xuzheng/ssl/swap_xuzheng.crt;
        ssl_certificate_key     /home/xuzheng/ssl/swap_xuzheng.key;
        server_name balanceador_xuzheng;

        ssl_session_cache       shared:SSL:1M;
        ssl_session_timeout     10m;


        access_log /var/log/nginx/balanceador_xuzheng.access.log;
        error_log /var/log/nginx/balanceador_xuzheng.error.log;
        root /var/www;

        location /
        {
                proxy_pass http://balanceo_xuzheng;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_http_version 1.1;
                proxy_set_header Connection "";
        }

}
```

4. Bibliografía

- https://www.openssl.org/docs/manmaster/man1/openssl-req.html
- https://linux.die.net/man/1/tar
- https://serverfault.com/questions/141773/what-is-archive-mode-in-rsync
- https://ss64.com/bash/rsync.html
- https://linux.die.net/man/1/rsync
- https://serverfault.com/questions/123629/run-task-every-90-minutes-with-cron